User's manual for AASeq*

Bruno Zanuttini
GREYC†, Université de Caen
Boulevard du Maréchal Juin
14 032 Caen Cedex, France
zanutti@info.unicaen.fr

Joël Henry
LBBM‡, Université de Caen
Esplanade de la Paix
14 032 Caen Cedex, France
joel.henry@unicaen.fr

May 2005

Contents

1 Presentation of AASeq **2**

2 Installation **3**

3 Usage **3**
3.1 Commands 3
3.2 An example 4
3.3 Important note 4
3.4 Errors and returned values 4
3.5 Command prompts 5

4 Command files and databases **5**
4.1 Syntax 5
4.2 Mandatory constraints and options 6
4.3 Other available constraints and options 7
4.4 Databases 11

5 Technical support **12**

*The version of AASeq at the time we write this manual is version 5.0. Therefore this is the version concerned here. Please visit regularly the website of AASeq at http://www.info.unicaen.fr/~zanutti/aaseq for upgrades.

†Groupe de Recherches en Informatique, Image, Automatique et Instrumentation de Caen

‡Laboratoire de Biologie et Biotechnologies Marines

1 Presentation of AASeq

AASeq is a software for helping *de novo* sequencing of peptides. Used together with a mass spectrometer, it allows to circumvent the difficulty of *de novo* sequencing due to the absence of databases of known sequences to which new spectra can be compared in traditional sequencing.

AASeq does not analyze MS spectra, neither does it give the sequence of a peptide. What it is able to do is to create databases of sequences that can be used to "feed" other softwares. More precisely, it is able to create a (virtual) database of all the sequences of amino acids that match given constraints. The constraints concern the mass of the sequences and, if desired, their size, their subsequences (N-terminal, C-terminal or internal), the number of certain amino acids etc. Consequently, if the peptide to be sequenced indeed satisfies these constraints, then its sequence will be in the database generated by AASeq, and consequently, if the MS/MS spectrum is good the sequence will be found out. All the databases generated by AASeq are in fasta format.

Thus AASeq can be used helpfully as soon as enough information has been collected about the peptide for the number of candidate sequences to be reasonable. This information can be obtained thanks to a mass spectrometer as concerns the approximate mass of the peptide, through testing of its MS/MS spectrum against random sequences (for instance generated by AARand[1]) as concerns size, subsequences and numbers of given amino acids, by intuition, from knowledge about the family of the peptide etc.

The use of AASeq can be summarized as follows:

1. Get a MS/MS spectrum of your peptide
2. Collect information about your peptide
3. Use this information as constraints for AASeq
4. Test the MS/MS spectrum against the sequences generated by AASeq.

The use of AASeq is thus completely independent from the software you use for analyzing your MS/MS spectra; you can use any one that can read sequences in fasta format.

Authors and conditions of use AASeq is developed at the University of Caen, France, by Joël Henry (laboratory *LBBM*, *IBFA*) and Bruno Zanuttini (laboratory *GREYC*). You are free to download and use it without any fee. You may also redistribute it for free or modify it under the terms of the *GNU General Public Licence*[2].

[1] Companion software of AASeq. See http://www.info.unicaen.fr/~zanutti/aaseq.
[2] Available at http://www.gnu.org/copyleft/gpl.html.

2 Installation

Installing and using AASeq does not require any special hardware configuration. As for operating systems, AASeq is available both for Linux and for Windows 95 or later. There is no other software requirement.

The standard installation of AASeq is composed of an executable file, named aaseq (Linux) or aaseq.exe (Windows), and of four text files. Only the executable one is necessary. The files have the following roles:

- aaseq(.exe): Executable file, i.e., the software itself;
- aasacids.txt: Database of the twenty common amino acids with their average masses;
- command.txt: An explanation of every constraint and option available for generation of sequences;
- database.txt: An example database of acids with syntax explanation, for use if you want to create your own database;
- template.txt: A template command file meant to be used as a basis for creating your own command files.

There is no particular requirement for the directories where AASeq must be installed. However, the simpler is certainly to create a new directory and to store the five files into it.

Installation indeed consists in copying the executable file and, if desired, the text files to this directory. However, if you do not want to install all the files in the same directory this will not alter AASeq's functioning. You can get the files from AASeq's website at http://www.info.unicaen.fr/~zanutti/aaseq, where precise directives are given for both Linux and Windows, or you can ask them to the authors. Once the files on your computer you are ready to use AASeq.

3 Usage

AASeq generates sequences from the options and constraints given in a "command file", which is simply a text file with a special syntax. In this section we assume that you have written this file, and we explain how to do that in next section.

3.1 Commands

The method for asking AASeq to generate the sequences that match the constraints and options specified in your command file is:

1. getting a command prompt and going to the right directory (see the end of this section if you do not know about command prompts);

2. typing `aaseq name-of-command-file` (and pressing `enter`).

If your command file is named `something.txt` then you will get the sequences in file `something.fasta` in the same directory. This file will contain all the desired sequences in `fasta` format and will be ready to be tested against your MS/MS spectrum using your favourite software for that purpose[3].

There are also two useful options for `AASeq`:

- `aaseq --version` displays version and license information,
- `aaseq --help` displays a quick help about basic usage.

3.2 An example

The most basic use consists in asking `AASeq` to generate all sequences with a given mass. To generate all sequences with a total molecular mass of $500Da$ plus or minus $1Da$ type the following into a file named, for instance, `seqs500.txt` in the directory where you installed `AASeq` (see below for the meaning of option `water-mass`):

```
[database] aasacids.txt
[molecular-mass] 500 ~ 1
[water-mass] 0
```

Save the file, then get a command prompt and go to the right directory (see below) and type `aaseq seqs500.txt`. Press `enter`; the desired sequences are now stored in file `seqs500.fasta` in the directory where you installed `AASeq`.

3.3 Important note

Do not use `AASeq` for, e.g., generating all sequences weighing between 500 and $1000Da$. The only thing you will get is a full disk! `AASeq` can generate sequences very quickly, but it is designed to generate every sequence it is asked to. It is your own responsibility to know in advance whether there will be a reasonable number of them, so that they can all be stored on your disk, but also so that your other softwares can handle all of them.

3.4 Errors and returned values

If an error occurs during reading of the command file or of the database of acids, or during the generation of sequences, a message will be displayed and

[3]Of course, those who are used to these mechanisms may add the path to `AASeq` to their path environment variable then run `AASeq` from any directory etc. However, note that the name of the database of acids, when relative, is evaluated with respect to the working directory, not to the directory that contains the command file where the database is specified. This is explained in more details in next section.

generation will be aborted. Messages are as explicit as possible and should guide you for correcting the error if it is your responsibility.

If generation went well `AASeq` returns 0, and in any other case it returns 1.

3.5 Command prompts

Because `AASeq` does not have any graphical interface (yet), running it requires that you get a so-called "command prompt". `Linux` users should be used to that; they need to launch a "terminal" (for instance an `xterm`) from their applications menu. As for `Windows` users, they need to launch a `Command Prompt` from their `Start` menu.

Both under `Linux` and under `Windows` you will then have to go to the directory where you installed `AASeq` with a `cd` command: Type `cd directory` in the command prompt, e.g., `cd ~/aaseq` (`Linux`), `cd c:\Program" "Files\AASeq`, `cd d:\MyPrograms\AASeq` (`Windows`) etc., and press `enter`. You will then be ready to use `AASeq` as explained above.

4 Command files and databases

`AASeq` uses text files as command files and databases of amino acids. You may create such text files by using any text editor, for instance `NotePad`, `Emacs`, `Vi` etc. Nevertheless, if you use editors such as `Word`, remember to save the file in "plain text" (.txt) format. Also name your command files `something.txt`.

Constraints and options available in command files are summarized in Table 1 on Page 13.

4.1 Syntax

A command file must contain one option or constraint per line, in the form `[ option-name ] option-value`. However, lines beginning with character `>` are not taken into account, which allows you to write comments in your file for remembering whatever you want when you read it again. Similarly, blank lines are not taken into account. The order of lines is not meaningful.

Masses are expressed in Daltons (Da), may it be for the mass of the sequences to generate or for differential modifications. The maximum precision is six decimal digits, and the maximum mass allowed is $4000Da$ (you may specify some more Daltons, but the result is not guaranteed; moreover, remind the note in previous section about the size of the output file). Thus you can specify masses between $0.000001Da$ and $3999.999999Da$. The same restrictions apply to the mass of each amino acid in the database.

The names of amino acids must be capital letters (`A-Z`) and must be the same in the database and in the command file. A sequence of amino acids is written as the sequence of all their names *without any space*.

Finally, note that spaces and case are not important except when explicitly mentionned below. However, once again do not split a constraint or option

([option-name] option-value) over two or more lines, and do not write several constraints or options on the same line.

The file named `command.txt` and distributed together with `AASeq` recalls the syntax of all options. When you want to create your own command files, you may also copy the file named `template.txt` and use it as a basis.

4.2 Mandatory constraints and options

There are three mandatory constraints and options: the database of amino acids, the mass of the sequence and the mass of water.

Database of amino acids The option name is `database` and its value is the name or path of the file containing the database of amino acids. The amino acids, together with their masses, specified in this file will be those composing the generated sequences. The syntax of database files is explained at the end of this section.

Here are some example lines specifying the database in a command file:

```
> Under Linux:
[database] ~/aaseq/aasacids.txt
> Under Windows:
[database] c:\Program Files\AASeq\aasacids.txt
[database] c:\Program Files\AASeq\myacids.txt
[database] c:\Documents and Settings\myacids.txt
```

You may also specify a relative path to the file (for instance `myacids.txt`). However, the absolute path will be evaluated relatively to the working directory (i.e., the directory from which you run `AASeq`) and not relatively to the directory which contains the command file.

Molecular mass The option name is `molecular-mass` and its value is a range for the mass of the sequences to generate (modified by the mass of water, see below). This range can be specified in two manners: either in the form `mass1 - mass2`, meaning that `AASeq` must generate sequences weighing between `mass1` and `mass2` (included), or in the form `mass ~ precision`, meaning that `AASeq` must generate sequences weighing `mass` up to `precision` (or, equivalently, between `mass - precision` and `mass + precision`).

Importantly, you must not give `AASeq` an m/z value; you must give the molecular mass of the sequence.

Here are some example lines; in this context we assume that the mass specified for water is 0 (see below):

```
> 123.456 Da up to 5.456 (i.e., between 118.0 and 128.912 Da):
[molecular-mass] 123.456 ~ 5.456
```

```
> Between 123.456 and 234.0 Da:
[molecular-mass] 123.456 - 234
```

Mass of water The mass of a peptide is usually measured including a water molecule. That is why `AASeq` automatically deduces the mass of this molecule from the masses you specify for sequences. Option `water-mass` lets you give this mass. Anyway, if you want to deduce it yourself or if your measures take only the masses of amino acids into account, and thus you wish to specify the exact mass of the sequences to generate (sum of the masses of their acids), then simply set the mass of water to 0.

Here are some example lines:

```
> The two following lines together mean (105.446=123.456-18.01):
> Sum of masses of acids in a sequence is 105.446 Da up to 5.456
[molecular-mass] 123.456 ~ 5.456
[water-mass] 18.01
> The two following, between 105.446 and 215.99 Da:
[molecular-mass] 123.456 - 234
[water-mass] 18.01
> The two following, between 123.456 and 234 Da:
[molecular-mass] 123.456 - 234
[water-mass] 0
```

4.3 Other available constraints and options

While the previous constraints and options are mandatory, the following are optional. Thus you may specify some of them or none at all. Importantly, the more constraints you specify, the less sequences you will obtain, thus speeding up future search. The available constraints and options concern: The size of the sequences, N- and C-terminal differential modifications, numbers of occurrences of amino acids and imposed subsequences.

Size of sequences This option allows you to constrain the generated sequences to those within a given size range. It is useless to specify this option if the values can be deduced from the mass of the sequences and the masses of the amino acids. Otherwise, you may specify a lower bound, an upper bound or both, in the form `[size] lower-bound - upper-bound`, omitting one of the bounds if desired; bounds are included.

Here are some example lines:

```
> Only generate sequences containing between 5 and 7 amino acids
[size] 5-7
> Only generate sequences containing at most 7 amino acids
[size] -7
```

```
> Only generate sequences containing at least 5 amino acids
[size] 5-
```

N- and C-Terminal modifications This option allows you to specify differential modifications on the mass of some amino acids when positionned at the N- or C-terminal extremity. The option value is `name-of-acid modification`, where `modification` is of the form `-mass` if the amino acid looses `mass` Daltons and of the form `+mass` if the amino acid gains `mass` Daltons. You can specify as many modifications as you desire, however the behaviour of `AASeq` is not defined if the same amino acid is subject to several N-terminal modifications (or to several C-terminal modifications), including the case when the peptide is amidated and an amino acid is subject to another C-terminal modification. Similary, the behaviour of `AASeq` is not defined if an amino acid is subject to both an N-terminal and a C-terminal modification and the sequence containing only this amino acid matches all other constraints; indeed, in this case the amino acid is both N-terminal and C-terminal.

Here are some example lines:

```
> When N-terminal, amino acid G gains 5.5 Da
[nter-modification] G +5.5
> When N-terminal, amino acid G looses 5.5 Da
[nter-modification] G -5.5
> When C-terminal, amino acid F gains 5.5 Da
[cter-modification] F +5.5
> When C-terminal, amino acid F looses 5.5 Da
[cter-modification] F -5.5
```

Two special differential modifications are available: amidation and pyroglutamate. Amidation induces the same mass loss for every acid in the database when C-terminal; you must specify both the fact that amidation occurs in your peptide, with option `amide` (without any value), and the mass loss, with option `amide-mass-loss` and value `mass-loss` (without `-` sign).

As for pyroglutamate, it concerns the often encountered mass loss of glutamine when N-terminal. You must specify that pyroglutamate occurs, with option `pyroglutamate` (without any option), the mass loss occurring, with option `pyroglutamate-mass-loss` and value `mass-loss` (without `-` sign), and the name of glutamine, with option `glutamine-name` and value `name`.

Here are some example lines:

```
> Every acid looses 0.984 Da when C-terminal
[amide]
[amide-mass-loss] 0.984
> Acid named Q looses 17.0265 Da when C-terminal
[pyroglutamate]
```

```
[pyroglutamate-mass-loss] 17.0265
[glutamine-name] Q
```

What makes `AASeq` take amidation into account is option `amide`: If your command file contains option `amide-mass-loss` but not option `amide` then the sequences will not be considered to be amidated. Similarly, if your command file contains option `pyroglutamate-mass-loss`, option `glutamine-name` or both but not option `pyroglutamate`, glutamine will be considered to weigh its normal mass even when N-terminal.

Numbers of occurrences of amino acids This constraint allows you to specify that a given amino acid occurs a given number of times in each sequence. You can specify a maximum number of occurrences, a minimum one or both, and you can do that for as many acids as you desire. For a given acid, this is specified by option `[acid] name-of-acid minimum-number - maximum-number`, omitting one of the bounds if desired; bounds are included. Note that in particular, this allows you to specify that a given acid cannot occur in the sequence, thanks to line `[acid] name-of-acid -0` (this can also be achieved by removing the acid from the database).

Here are some example lines:

```
> Acid L occurs at least two and at most four times
[acid] L 2-4
> Acid L occurs at most four times
[acid] L -4
> Acid L occurs at least two times
[acid] L 2-
> Acid L does not occur at all
[acid] L -0
```

Subsequences This last group of options allows you to impose some subsequences to the sequences to be generated. First of all, you can specify the possible N-terminal and/or C-terminal subsequences with options `nterminal` and `cterminal`. At most one value can be specified for each of these options (because the sequence of a peptide cannot begin or end with two different subsequences), but the value is a list of possible subsequences. More precisely, N-terminal possible subsequences are given in the form `[nterminal] sub1 | sub2 | sub3...`, and similarly for C-terminal subsequences.

Here are some example lines:

```
> The sequence of the peptide begins either with GNL or with GNI
[nterminal] GNL | GNI
> The sequence of the peptide begins with GNL
```

```
[nterminal] GNL
> The sequence of the peptide ends with RF or FR
[cterminal] RF | FR
> The sequence of the peptide ends with RF
[cterminal] RF
```

Importantly, subsequences may overlap; for instance, if you require `GNL` as an N-terminal subsequence and `LFRF` as a C-terminal subsequence, then sequence `GNLFRF` is considered to match both constraints together.

You can also specify subsequences that are not necessarily N-terminal or C-terminal. This can be done combining the following precisions: Whether you know the order of the subsequence and whether you know its position from the N-terminal or C-terminal extremity.

Similarly to the case of N-terminal and C-terminal subsequences, you may specify various possibilities: e.g., `[subsequences] sub1 | sub2` means that your peptide contains either `sub1` or `sub2` (or both). But you may also specify several such constraints: e.g., specifying both `[subsequences] sub1 | sub2` and `[subsequences] sub3 | sub4 | sub5` means that your peptide contains either `sub1` or `sub2`, but also contains either `sub3` or `sub4` or `sub5`.

As for precisions, they are given inside parentheses after the concerned subsequence. If the subsequence must appear in the exact order in which it is given, then nothing has to be specified. Otherwise, simply specify `u` (for "unordered") inside the parentheses; this means that the subsequence must occur in the sequences to generate, but maybe in a different order.

Finally, if you know the position of the subsequence from the N-terminal extremity, specify `N pos` inside the parentheses (meaning there are `pos` $-$ 1 acids in the sequence before the beginning of the subsequence); if you know the position of the subsequence from the C-terminal extremity, specify `C pos` (`pos` $-$ 1 acids after the end of the subsequence); finally, if you do not know the position, do not specify anything. You cannot specify a position from both the N-terminal and the C-terminal extremities. Also note that a subsequence specified with option `subsequence` may occur at the N-terminal or C-terminal extremity. Finally, if two precisions are given, separate them by a comma.

All together, here are all the possible precisions for a subsequence:

- `sub` or `sub ()`: The subsequence occurs at any position in the sequences, in the order in which it is given;
- `sub (u)`: The subsequence occurs at any position, but maybe in a different order than that in which it is given; for instance, `IRF (u)` means that either `IRF` or `IFR` or `FIR` or `FRI` or `RIF` or `RFI` must occur somewhere in the sequences;
- `sub (N pos)`: The subsequence occurs after exactly `pos` $-$ 1 acids in the sequences, and in the order in which it is given;

- `sub (C pos)`: The subsequence is followed by exactly `pos` -1 acids in the sequences, and in the order in which it is given;
- `sub (u,N pos)` or `sub (N pos,u)`: The subsequence occurs after exactly `pos` -1 acids in the sequences, but maybe in a different order;
- `sub (u,C pos)` or `sub (C pos,u)`: The subsequence is followed by exactly `pos` -1 acids in the sequences, but maybe in a different order.

Once again, subsequences may overlap, and also overlap with N-terminal and C-terminal required subsequences. For instance, sequence `GNLFRF` is considered to begin with `GNL`, to end with `LFRF`, to contain `FL (u)` and to contain `NLF (N2)` at the same time.

Here is finally a whole example of subsequences specification:

```
> The following lines all together impose that each generated
> sequence satisfies the three following constraints:
> (i)   it contains either KLF or RF or FR
> (ii)  it begins with xNL, where x is any acid
> (iii) either it ends with RKxxxx or it ends with KRxxxx or it
>       contains ALI or it contains WS
[subsequences] KLF | RF (u)
[subsequences] NL (N2)
[subsequences] RK (u,C5) | ALI | WS
```

4.4 Databases

As previously evoked, most rules that apply to the syntax of command files also apply to the syntax of database files: Acid names must be capital letters, masses can range from 0.000001 to $3999.999999Da$, lines beginning with character `>` and blank lines are ignored, and there must be one acid specified per line.

The syntax for one acid is simply `acid-name mass`; for instance:

```
> The database contains an acid named Y and weighing 128.01 Da
Y 128.01
> The database contains an acid named Z and weighing 82.00 Da
Y 82
```

The file named `aasacids.txt` and distributed together with `AASeq` contains a standard database of the twenty common amino acids together with their standard names and their average masses with a precision of four decimal digits.

If you want to create your own database, you can proceed by copying the file named `database.txt` and using it as a basis.

5 Technical support

Technical support can be obtained from Bruno Zanuttini (current e-mail address: `zanutti@info.unicaen.fr`). In case of error or observed bug, please communicate the corresponding command and database files as well as everything displayed by `AASeq`.

As already evoked, there is a website dedicated to `AASeq` (and `AARand`); this site is currently located at `http://www.info.unicaen.fr/~zanutti/aaseq`. New versions will be published there as well as known bugs, if any. You can also register there as an `AASeq` user so that we can support you efficiently.

Finally, according to the *GNU General Public Licence* you are allowed to modify and redistribute `AASeq`. The source files can be obtained from the website and explanations can be asked to Bruno Zanuttini.

Option	Example values	Meaning
`database`	`c:\AASeq\myacids.txt`	database file
`molecular-mass`	`123.456 ~ 5.456`	mass of seq. $= 123.456Da$ up to 5.456
	`123.456 - 234`	mass of seq. $\geq 123.456Da$ and $\leq 234Da$
`water-mass`	`18.01`	$\sum$ masses of acids = molecular mass $-18.01Da$
`size`	`5-7`	nb. acids in seq. ≥ 5 and ≤ 7
	`-7`	nb. acids in seq. ≤ 7
	`5-`	nb. acids in seq. ≥ 5
`nter-modification`	`G +5.5`	when N-terminal, G gains $5.5Da$
	`G -5.5`	when N-terminal, G looses $5.5Da$
`cter-modification`	`F +5.5`	when C-terminal, F gains $5.5Da$
	`F -5.5`	when C-terminal, F looses $5.5Da$
`amide`	no value	all acids loose the same mass when C-ter.
`amide-mass-loss`	`0.984`	if amidation occurs, mass loss is 0.984
`pyroglutamate`	no value	glutamine looses some mass when N-ter.
`pyroglutamate-mass-loss`	`17.0265`	if pyroglutamate occurs, mass loss is 17.0265
`glutamine-name`	`Q`	pyroglutamate concerns acid symbol Q
`acid`	`L 2-4`	nb. of L in seq. is ≥ 2 and ≤ 4
	`L -4`	nb. of L in seq. is ≤ 4
	`L 2-`	nb. of L in seq. is ≥ 2
`nterminal`	`GNL | GNI`	seq. begins with GNL or GNI
`cterminal`	`RF | FR`	seq. ends with RF or FR
`subsequences`	`KLF | RF (u)`	seq. contains KLF or RF or FR
	`NL (N2)`	seq. contains NL in 2nd position
	`RK (u,C5) | ALI | WS`	seq.$=\ldots RKxxxx, \ldots KRxxxx, \ldots ALI \ldots$ or $\ldots WS \ldots$

Table 1: List of all constraints and options available in command files